

<u>PARAMETER</u>	<u>DESCRIPTION</u>	<u>WHERE TO FIND</u>	<u>LINE OF CODE TO MODIFY</u>	<u>MODIFIED LINE OF CODE</u>	<u>INSTRUCTION & NOTES</u>
<u>Machine Name</u>		<u>Marlin/ Configuration.h</u>	<u>#define MACHINE_NAME "My 3D Printer"</u>	<u>#define MACHINE_NAME "CASI Mk3"</u>	
<u>Config Author</u>		<u>Marlin/ Configuration.h</u>	<u>#define STRING_CONFIG_H_AUTHOR "(thisiskeithb, Ender-3 Pro)"</u>	<u>#define STRING_CONFIG_H_AUTHOR "TechBuilder"</u>	
<u>Enable CoreXY</u>	<u>The Marlin template is preset to the standard cartesian model. This part enables CoreXY mode.</u>	<u>Marlin/ Configuration.h</u>	<u>~~c//#define COREXY~~</u>	<u>#define COREXY</u>	<u>Just uncomment the code to enable CoreXY mode instead the standard cartesian. (Uncommenting is done by removing the // before the #define)</u>
<u>Build Area Size</u>	<u>Your heatbed's size determines the buildplate's size (your printer's workable XY print area) while your Z axis defines the maximum height of 3DP object you can print.</u>	<u>Marlin/ Configuration.h</u>	<u>#define X_BED_SIZE 235</u>	<u>#define X_BED_SIZE 235</u>	<u>In the Marlin template I used the heatbed size is already properly set to my actual heatbed's dimensions (235x235mm). In case you decide to use a smaller heatbed, kindly adjust these accordingly for the X & Y.</u> <u>The Z height is dictated by the maximum distance your heatbed is from your extruder nozzle. To get this distance, home your 3DP first. Move your Z-axis upwards until the you hear the stepper</u>
			<u>#define Y_BED_SIZE 235</u>	<u>#define Y_BED_SIZE 235</u>	
			<u>#define Z_MAX_POS 250</u>	<u>#define Z_MAX_POS 220</u>	

					<u>motors skip steps. At this point measure the distance between the nozzle and heatbed. Mine was approximately 223mm. I recommend inputting a lower value as a safety margin, thus the 220.</u>
<u>E-Steps</u>	<u>E-steps denote the number of steps required for a stepper motor to traverse one millimeter. This calibration factor is essential for converting radial values to rectilinear values within the 3D printer's motion control system. By accurately determining the E-steps for each stepper motor, the 3D printer achieves precise control over movements, contributing to the accuracy and quality of the printed objects."</u> <u>This part of the</u>	<u>Marlin/ Configuration.h</u>	<u>#define DEFAULT_AXIS_STEPS_PER_UNIT { 80, 80, 400, 93 }</u>	<u>~~#define DEFAULT_AXIS_STEPS_PER_UNIT { 80, 80, 640, 139 }~~</u>	<u>139</u> This line of code refers to the E-steps for the {X, Y, Z, Extruder} stepper motors. Since we are using a CoreXY mechanism it is basically {M1, M2, Z, Extruder}_. <u>X & Y Axis E-steps: can be determined by using this formula: Esteps = 3200/(2*Pulley Teeth)</u> <u>I'm using a 20T (20 tooth pulley) on my M1 and M2 steppers on my CoreXY printer, 20T gives me an E-step of 80. Hence the 80 for the X and the 80 for the Y.</u> <u>Z Axis E-st</u>

	<u>guide will also explain how to compute for them. Take note that the computation and declaration of E-steps here serves as a baseline value. To further increase your printer's accuracy, kindly search for "3D Printer E-step Calibration". There are tons of video tutorials guides that will help you through the calibration process.</u>				<pre>#define DEFAULT_AXIS_STEPS</pre>
--	--	--	--	--	---------------------------------------

					<u>eps: can be determined by using this formula:</u> <u>Esteps = 3200/Lead Screw Pitch</u> <u>My Z-axis uses a modified/ repurposed linear actuator, just a fancy industrial version of the leadscrew & rod assembly of most 3D printers. Regardless, you just have to measure the spacing between your leadscrew to find the pitch. Leadscrews variations often come in increments of 1m or 0.5mm (ex: 1mm, 2mm, 2.5mm, 6mm, 8mm). No need for calipers, your ruler will suffice. Mine was determined to have a 2.5mm pitch, the formula gives me an E-set of 640.</u> <u>Extruder E-steps:</u> <u>Bowden or direct drive?</u> <u>Same thing... I wouldn't include the math formula for it since,</u> <u>most extruder manufacturers specify the E-steps for it. I'm using a Generic Dual</u>
--	--	--	--	--	---

					<u>Gear Drive Extruder; it is known to have an E-Step value of 139. If you opt to use the generic single gear extruder from CR10s or Ender 3s; use the value of 93.</u>
<u>Feed Rate</u>		<u>Marlin/ Configuration.h</u>	<u>#define DEFAULT_MAX_FEEDRATE { 200, 200, 5, 25 }</u>	<u>#define DEFAULT_MAX_FEEDRATE { 1000, 1000, 10, 1000 }</u>	
<u>Acceleration</u>		<u>Marlin/ Configuration.h</u>	<u>#define DEFAULT_MAX_ACCELERATION { 1000, 1000, 100, 10000 }</u>	<u>#define DEFAULT_MAX_ACCELERATION { 20000, 20000, 100, 10000 }</u>	
			<u>#define DEFAULT_ACCELERATION 500</u>	<u>#define DEFAULT_ACCELERATION 20000</u>	
			<u>#define DEFAULT_TRAVEL_ACCELERATION 500</u>	<u>#define DEFAULT_TRAVEL_ACCELERATION 20000</u>	
			<u>#define DEFAULT_RETRACT_ACCELERATION 500</u>	<u>#define DEFAULT_RETRACT_ACCELERATION 700</u>	
<u>Jerk</u>		<u>Marlin/ Configuration.h</u>	<u>#define DEFAULT_EJERK 5.0</u>	<u>#define DEFAULT_EJERK 15.0</u>	
<u>Stepper Motor Direction</u>		<u>Marlin/ Configuration.h</u>	<u>#define INVERT_X_DIR true</u>	<u>#define INVERT_X_DIR false</u>	<u>You can test the stepper motor direction by using the Move Axis on the LCD menu.</u> <u>In “Move X-Axis”, scrolling up to a positive distance should move the printhead to the right.</u>
			<u>#define INVERT_Y_DIR true</u>	<u>#define INVERT_Y_DIR false</u>	
			<u>#define INVERT_Z_DIR false</u>	<u>#define INVERT_Z_DIR true</u>	
			<u>#define INVERT_E0_DIR true</u>	<u>#define INVERT_E0_DIR true</u>	

					<u>In “Move Y-Axis”, scrolling up to a positive distance should move the printhead to the rear.</u> <u>In “Move Z-Axis”, scrolling up to a positive distance should drop the heatbed lower.</u> <u>Note: If it is moving in the opposite direction, invert the code’s logic (if true, make it false - if false, make it true). This is how I came up with my config. It may not apply to all since not all stepper motors are wired the same way.</u>
<u>Endstop Logic Setup</u>		<u>Marlin/ Configuration.h</u>	<u>#define ENDSTOPPULLUPS</u>	<u>#define ENDSTOPPULLUPS</u>	<u>In my build all endstops are wired to the C and NO of the limit switches. This requires all the endstop trigger logic to be inverted.</u>
			<u>#define X_MIN_ENDSTOP_INVERTING false</u> <u>#define Y_MIN_ENDSTOP_INVERTING false</u> <u>#define Z_MIN_ENDSTOP_INVERTING false</u>	<u>#define X_MIN_ENDSTOP_INVERTING true</u> <u>#define Y_MIN_ENDSTOP_INVERTING true</u> <u>#define Z_MIN_ENDSTOP_INVERTING true</u>	
<u>Endstop Location (Limit Switch)</u>	<u>This teaches you how to configure the firmware whenever you decide to relocate the endstop (limit switch) placement locations. The</u>	<u>Marlin/ Configuration.h</u>	<u>#define X_HOME_DIR -1</u>	<u>#define X_HOME_DIR -1</u>	<u>Marlin states that 1 = Max and 0 = Min. Selecting max means you limit switch is located in the upper portion of the specified axis. The guide below specifically applies to</u>
			<u>#define Y_HOME_DIR -1</u>	<u>#define Y_HOME_DIR 1</u>	
			<u>#define Z_HOME_DIR -1</u>	<u>#define Z_HOME_DIR -1</u>	

	<u>Marlin template I am using is using a different location for the Endstops as to what my actual build is using.</u>				<u>my CoreXY.</u> <u>Y-Endstop:</u> <u>□ Mounted at the left = -1</u> <u>□ Mounted at the right = 1</u> <u>X-Endstop:</u> <u>□ Mounted in Front = -1</u> <u>□ Mounted at the Rear = 1</u> <u>Z-Endstop:</u> <u>□ Mounted on top = -1</u> <u>□ Mounted at the bottom = 1</u>
<u>Stepper Current Limit (TMC Driver)</u>	<u>You must adjust the stepper motor current on your controller to maximize the torque of the NEMA 17 stepper motors you have. If you are using the BigTreeTech SKR Mini E3 V3, you can adjust the</u>	<u>Marlin/ Configuration_adv.h</u>	<u>#define X_CURRENT 580</u>	<u>#define X_CURRENT 1273</u>	<u>Check your NEMA 17 stepper motor's datasheet, the peak current is usually specified. The code in Marlin for the TMC driver current is notated in RMS. That mean the peak current from the datasheet needs to be divided by $\sqrt{2}$ before inputting to Marlin. For</u>
			<u>#define Y_CURRENT 580</u>	<u>#define Y_CURRENT 1273</u>	
			<u>#define Z_CURRENT 580</u>	<u>#define Z_CURRENT 1273</u>	
			<u>#define E0_CURRENT 650</u>	<u>#define E0_CURRENT 750</u>	

	<u>stepper current limit via firmware.</u>				<u>example, my stepper is rated at 1.8A peak; 1.8A/√2 = 1.273A RMS. Use these values for the X, Y & Z stepper motors. Meanwhile I recommend setting the current of the extruder motor to ½ of the XYZ.</u>
<u>LCD (FSYTEC 12864 RGB)</u>	<u>By default, the Marlin template is set to work with the Reprap Discount LCD (Ones often used for Repraps, CR10 and Ender 3s). Use this if you decide to go with the FSYTEC 12864 RGB LCD.</u>	<u>Marlin/src/lcd /dogm / marlinui_DOGM .cpp</u>	<u>//#define REPRAP_DISCOUNT_FULL_GRAPHIC_SMART_CONTROLLER</u>	<u>//#define REPRAP_DISCOUNT_FULL_GRAPHIC_SMART_CONTROLLER</u>	
			<u>//#define FYSETC_MINI_12864_2_1</u>	<u>#define FYSETC_MINI_12864_2_1</u>	
			<u>#define NEOPIXEL_TYPE NEO_GRB</u>	<u>#define NEOPIXEL_TYPE NEO_RGB</u>	
			<u>#define NEOPIXEL_PIXELS 10</u>	<u>#define NEOPIXEL_PIXELS 3</u>	
			<u>#define NEOPIXEL_BRIGHTNESS 127</u>	<u>#define NEOPIXEL_BRIGHTNESS 255</u>	
			<u>u8g.setContrast(contrast);</u>	<u>u8g.setContrast(255);</u>	
<u>Hotend Maximum Temp Limit</u>	<u>This parameter dictates the maximum allowable hotend temperature your 3D printer can print on. Exotic filaments may require high temps. Kindly refer to your hotend model's datasheet to fully utilize its full potential.</u>		<u>#define HEATER_0_MAXTEMP 275</u>	<u>#define HEATER_0_MAXTEMP 300</u>	<u>One of my machines uses an E3D Volcano Clone capable up to 300°C. And my other printer uses a Trianglelab CHC Pro Volcano Clone capable of 320°C. Set the limit according to what your hotend can reach.</u>

Enable Z-offset Adjustment (For Non-Bltouch Users)		Marlin/Configuration_adv.h	//#define BABYSTEP_ZPROBE_OFFSET	#define BABYSTEP_ZPROBE_OFFSET	
Filament Runout	Filament runout lets you add a sensor that detects if your filament runs out. This pauses your print and lets you load a new filament. You can add a sensor to your BTT SKR Mini E3 V3's E-stop port.	Marlin/Marlin/Configuration.h	//#define FILAMENT_RUNOUT_SENSOR	#define FILAMENT_RUNOUT_SENSOR	In my video tutorial, my Casi MK3 build is not using a filament runout sensor, however my undocumented Casi MK2 build does use one. I'll make a video tutorial on my clog-free design implementation used on the MK2. For now, you can use any 3D printable limit switch runout sensor design. Just solder a wire from the C and NC pins of the limit switch to your BTT SKR Mini's E-stop port's GND & PC15 pins. In this setup, when the limit switch is not pressed (absence of filament), the runout is triggered. NOTE: This specific config, does not trigger filament runout when a runout sensor is not connected. You can
			#define FIL_RUNOUT_STATE LOW	#define FIL_RUNOUT_STATE LOW	

					leave it enabled even if you don't have a runout sensor yet.
BL-Touch		Marlin/Configuration.h	//#define BLTOUCH	#define BLTOUCH	My Casi MK3 build in the video tutorial does not use BLtouch. My MK2 however, uses BLtouch for probing. In case you decide to add BL-touch, just follow these configs.
			//#define Z_MIN_PROBE_USES_Z_MIN_ENDSTOP_PIN	#define Z_MIN_PROBE_USES_Z_MIN_ENDSTOP_PIN	
			#define NOZZLE_TO_PROBE_OFFSET { -40, -10, -1.85 }	#define NOZZLE_TO_PROBE_OFFSET { 10, 2, -5.30 }	
			#define MIN_SOFTWARE_ENDSTOP_Z	//#define MIN_SOFTWARE_ENDSTOP_Z	
			//#define Z_SAFE_HOMING	#define Z_SAFE_HOMING	
			//#define AUTO_BED_LEVELING_BILINEAR	#define AUTO_BED_LEVELING_BILINEAR	
			#define MESH_BED_LEVELING	//#define MESH_BED_LEVELING	
		Marlin/Configuration_adv.h	//#define BLTOUCH_DELAY 500	#define BLTOUCH_DELAY 500	
			#define BLTOUCH_FORCE_SW_MODE	#define BLTOUCH_FORCE_SW_MODE	
			//#define PROBE_OFFSET_WIZARD	#define PROBE_OFFSET_WIZARD	
Temperature Sensor Type Selection	Most 3D printer components supplied online comes with EPCOS 3950 NTC Thermistor sensors (especially the heatbed). Hotend sensors may vary. This allows you to	Marlin/Configuration_adv.h	#define TEMP_SENSOR_0 1	#define TEMP_SENSOR_0 1	Heatbeds often use 3950 EPCOS NTC Thermistors, so leave it be in the config. For your hotend, however, I suggest checking your supplier's page or datasheet to see what thermistor was used.
			#define TEMP_SENSOR_BED 1	#define TEMP_SENSOR_BED 1	

	<u>select your NTC thermistor model for better accuracy.</u>				<u>To know which value to input do a CTRL+F on “@section temperature” a table will show the different Thermistor models with their corresponding part code for the config.</u> <u>My Casi MK3 uses the standard 3950s for both hotend and heatbed, so I left them all as “1”. My Casi MK2 uses a Volcano CHC with a 104NT-4 Thermistors, on that machine my value is “5” for the hotend and “1” for the heatbed.</u>
--	--	--	--	--	---

PARAMETER	DESCRIPTION	WHERE TO FIND	LINE OF CODE TO MODIFY	MODIFIED LINE OF CODE	INSTRUCTION & NOTES
Machine Name		Marlin/ Configuration.h	#define MACHINE_NAME "My 3D Printer"	#define MACHINE_NAME "Casi Mk3"	
Config Author		Marlin/ Configuration.h	#define STRING_CONFIG_H_AUTHOR "(thisiskeithb, Ender-3 Pro)"	#define STRING_CONFIG_H_AUTHOR "TechBuilder"	
Enable CoreXY	The Marlin template is preset to the standard cartesian model. This part enables CoreXY mode.	Marlin/ Configuration.h	//#define COREXY	#define COREXY	Just uncomment the code to enable CoreXY mode instead the standard cartesian. (Uncommenting is done by removing the // before the #define)

Build Area Size	Your heatbed's size determines the buildplate's size (your printer's workable XY print area) while your Z axis defines the maximum height of 3DP object you can print.	Marlin/ Configuration.h	#define X_BED_SIZE 235	#define X_BED_SIZE 235	In the Marlin template I used the heatbed size is already properly set to my actual heatbed's dimensions (235x235mm). In case you decide to use a smaller heatbed, kindly adjust these accordingly for the X & Y. The Z height is dictated by the maximum distance your heatbed is from your extruder nozzle. To get this distance, home your 3DP first. Move your Z-axis upwards until the you hear the stepper motors skip steps. At this point measure the distance between the nozzle and heatbed. Mine was approximately 223mm. I recommend inputting a lower value as a safety margin, thus the 220.
			#define Y_BED_SIZE 235	#define Y_BED_SIZE 235	
			#define Z_MAX_POS 250	#define Z_MAX_POS 220	

E-Steps	E-steps denote the number of steps required for a stepper motor to traverse one millimeter. This calibration factor is essential for converting radial values to rectilinear values within the 3D printer's motion control system. By accurately determining the E-steps for each stepper motor, the 3D printer achieves precise control over movements, contributing to the accuracy and quality of the printed objects." This part of the guide will also explain how to compute for them. Take note that the computation and declaration of E-steps here serves as a baseline value. To further increase your printer's accuracy, kindly search for "3D Printer E-step Calibration". There are tons of video	Marlin/ Configuration.h	<pre>#define DEFAULT_AXIS_STEPS_PER_UNIT { 80, 80, 400, 93 }</pre>	<pre>#define DEFAULT_AXIS_STEPS_PER_UNIT { 80, 80, 640, 139 }</pre>	This line of code refers to the E-steps for the {X, Y, Z, Extruder} stepper motors. Since we are using a CoreXY mechanism it is basically {M1, M2, Z, Extruder}. X & Y Axis E-steps: can be determined by using this formula: $Esteps = \frac{3200}{2 * Pulley\ Teeth}$ I'm using a 20T (20 tooth pulley) on my M1 and M2 steppers on my CoreXY printer, 20T gives me an E-step of 80. Hence the 80 for the X and the 80 for the Y. Z Axis E-steps: can be determined by using this formula: $Esteps = \frac{3200}{Lead\ Screw\ Pitch}$ My Z-axis uses a modified/ repurposed linear actuator, just a fancy industrial version of the leadscrew & rod assembly of most 3D printers. Regardless, you just have to measure the spacing between your leadscrew to find the pitch. Leadscrews variations often come in
---------	---	----------------------------	--	---	---

	tutorials guides that will help you through the calibration process.				increments of 1m or 0.5mm (ex: 1mm, 2mm, 2.5mm, 6mm, 8mm). No need for calipers, your ruler will suffice. Mine was determined to have a 2.5mm pitch, the formula gives me an E-set of 640. Extruder E-steps:- Bowden or direct drive? Same thing... I wouldn't include the math formula for it since, most extruder manufacturers specify the E-steps for it. I'm using a Generic Dual Gear Drive Extruder; it is known to have an E-Step value of 139. If you opt to use the generic single gear extruder from CR10s or Ender 3s; use the value of 93.
Feed Rate		Marlin/ Configuration.h	#define DEFAULT_MAX_FEEDRATE { 200, 200, 5, 25 }	#define DEFAULT_MAX_FEEDRATE { 1000, 1000, 10, 1000 }	
Acceleration		Marlin/ Configuration.h	#define DEFAULT_MAX_ACCELERATION { 1000, 1000, 100, 10000 }	#define DEFAULT_MAX_ACCELERATION { 20000, 20000, 100, 10000 }	
			#define DEFAULT_ACCELERATION 500	#define DEFAULT_ACCELERATION 20000	
			#define DEFAULT_TRAVEL_ACCELERATION 500	#define DEFAULT_TRAVEL_ACCELERATION 20000	
			#define DEFAULT_RETRACT_ACCELERATION 500	#define DEFAULT_RETRACT_ACCELERATION 700	
Jerk		Marlin/	#define DEFAULT_EJERK 5.0	#define DEFAULT_EJERK 15.0	

		Configuration.h			
Stepper Motor Direction		Marlin/ Configuration.h	#define INVERT_X_DIR true	#define INVERT_X_DIR false	You can test the stepper motor direction by using the Move Axis on the LCD menu. In “Move X-Axis”, scrolling up to a positive distance should move the printhead to the right. In “Move Y-Axis”, scrolling up to a positive distance should move the printhead to the rear. In “Move Z-Axis”, scrolling up to a positive distance should drop the heatbed lower. Note: If it is moving in the opposite direction, invert the code’s logic (if true, make it false—if false, make it true). This is how I came up with my config. It may not apply to all since not all stepper motors are wired the same way.
			#define INVERT_Y_DIR true	#define INVERT_Y_DIR false	
			#define INVERT_Z_DIR false	#define INVERT_Z_DIR true	
			#define INVERT_E0_DIR true	#define INVERT_E0_DIR true	
Endstop Logic Setup		Marlin/ Configuration.h	#define ENDSTOPPULLUPS	#define ENDSTOPPULLUPS	In my build all endstops are wired to the C and NO of the limit switches. This requires all the endstop trigger logic to be
			#define X_MIN_ENDSTOP_INVERTING false #define Y_MIN_ENDSTOP_INVERTING false	#define X_MIN_ENDSTOP_INVERTING true #define Y_MIN_ENDSTOP_INVERTING true #define Z_MIN_ENDSTOP_INVERTING true	

			#define Z_MIN_ENDSTOP_INVERTING false		inverted.
Endstop Location (Limit Switch)	This teaches you how to configure the firmware whenever you decide to relocate the endstop (limit switch) placement locations. The Marlin template I am using is using a different location for the Endstops as to what my actual build is using.	Marlin/ Configuration.h	#define X_HOME_DIR 1	#define X_HOME_DIR 1	Marlin states that 1 = Max and 0 = Min. Selecting max means you limit switch is located in the upper portion of the specified axis. The guide below specifically applies to my CoreXY. Y-Endstop: ☐ Mounted at the left = 1 ☐ Mounted at the right = 1 X-Endstop: ☐ Mounted in Front = 1 ☐ Mounted at the Rear = 1 Z-Endstop: ☐ Mounted on top = 1 ☐ Mounted at the bottom = 1
			#define Y_HOME_DIR 1	#define Y_HOME_DIR 1	
			#define Z_HOME_DIR 1	#define Z_HOME_DIR 1	
Stepper Current Limit (TMC Driver)	You must adjust the stepper motor current on your controller to maximize the torque of the NEMA 17 stepper motors you have. If you are	Marlin/ Configuration_adv.h	#define X_CURRENT 580	#define X_CURRENT 1273	Check your NEMA 17 stepper motor's datasheet, the peak current is usually specified. The code in Marlin for the TMC driver current is notated in RMS. That mean the peak
			#define Y_CURRENT 580	#define Y_CURRENT 1273	
			#define Z_CURRENT 580	#define Z_CURRENT 1273	
			#define E0_CURRENT 650	#define E0_CURRENT 750	

	using the BigTreeTech SKR Mini E3 V3, you can adjust the stepper current limit via firmware.				current from the datasheet needs to be divided by $\sqrt{2}$ before inputting to Marlin. For example, my stepper is rated at 1.8A peak; $1.8A/\sqrt{2} = 1.273A$ RMS. Use these values for the X, Y & Z stepper motors. Meanwhile I recommend setting the current of the extruder motor to $\frac{1}{2}$ of the XYZ.
LCD (FSYTEC 12864 RGB)	By default, the Marlin template is set to work with the Reprap Discount LCD (Ones often used for Repraps, CR10 and Ender 3s). Use this if you decide to go with the FSYTEC 12864 RGB LCD.	Marlin/src/lcd /dogm / marlinui_DOGM.c pp	##define REPRAP_DISCOUNT_FULL_GRAPHIC_SMART_CONTROLLER	##define REPRAP_DISCOUNT_FULL_GRAPHIC_SMART_CONTROLLER	
			##define FYSETC_MINI_12864_2_1	#define FYSETC_MINI_12864_2_1	
			#define NEOPIXEL_TYPE NEO_GRB	#define NEOPIXEL_TYPE NEO_RGB	
			#define NEOPIXEL_PIXELS 10	#define NEOPIXEL_PIXELS 3	
			#define NEOPIXEL_BRIGHTNESS 127	#define NEOPIXEL_BRIGHTNESS 255	
			u8g.setContrast(contrast);	u8g.setContrast(255);	
Hotend Maximum Temp Limit	This parameter dictates the maximum allowable hotend temperature your 3D printer can print on. Exotic filaments may require high temps. Kindly refer to your hotend model's datasheet to fully utilize its full potential.		#define HEATER_0_MAXTEMP 275	#define HEATER_0_MAXTEMP 300	One of my machines uses an E3D Volcano Clone capable up to 300°C. And my other printer uses a Trianglelab CHC Pro Volcano Clone capable of 320°C. Set the limit according to what your hotend can reach.
Enable Z-offset Adjustment		Marlin/ Configuration_ad	##define BABYSTEP_ZPROBE_OFFSET	#define BABYSTEP_ZPROBE_OFFSET	

(For Non-Bltouch Users)		v.h			
Filament Runout	Filament runout lets you add a sensor that detects if your filament runs out. This pauses your print and lets you load a new filament. You can add a sensor to your BTT SKR Mini E3 V3's E-stop port.	Marlin/ Marlin/ Configuration.h	//#define FILAMENT_RUNOUT_SENSOR	#define FILAMENT_RUNOUT_SENSOR	In my video tutorial, my Casi MK3 build is not using a filament runout sensor, however my undocumented Casi MK2 build does use one. I'll make a video tutorial on my clog-free design implementation used on the MK2. For now, you can use any 3D printable limit switch runout sensor design. Just solder a wire from the C and NC pins of the limit switch to your BTT SKR Mini's E-stop port's GND & PC15 pins. In this setup, when the limit switch is not pressed (absence of filament), the runout is triggered. NOTE: This specific config, does not trigger filament runout when a runout sensor is not connected. You can leave it enabled even if you don't have a runout sensor yet.
			#define FIL_RUNOUT_STATE LOW	#define FIL_RUNOUT_STATE LOW	
BL Touch		Marlin/ Configuration.h	//#define BLTOUCH	#define BLTOUCH	My Casi MK3 build in the video tutorial does not use BLtouch. My MK2 however, uses BLtouch for probing.
			//#define	#define	

			Z_MIN_PROBE_USES_Z_MIN_ENDSTOP_PIN	Z_MIN_PROBE_USES_Z_MIN_ENDSTOP_PIN	In case you decide to add BL touch, just follow these configs.
			~~#define NOZZLE_TO_PROBE_OFFSET { -40, -10, -1.85 }~~	~~#define NOZZLE_TO_PROBE_OFFSET { 10, 2, -5.30 }~~	
			~~#define MIN_SOFTWARE_ENDSTOP_Z~~	~~##define MIN_SOFTWARE_ENDSTOP_Z~~	
			~~##define Z_SAFE_HOMING~~	~~#define Z_SAFE_HOMING~~	
			~~##define AUTO_BED_LEVELING_BILINEAR~~	~~#define AUTO_BED_LEVELING_BILINEAR~~	
			~~#define MESH_BED_LEVELING~~	~~##define MESH_BED_LEVELING~~	
		Marlin/ Configuration_adv.h	~~##define BLTOUCH_DELAY 500~~	~~#define BLTOUCH_DELAY 500~~	
			~~#define BLTOUCH_FORCE_SW_MODE~~	~~#define BLTOUCH_FORCE_SW_MODE~~	
			~~##define PROBE_OFFSET_WIZARD~~	~~#define PROBE_OFFSET_WIZARD~~	
			~~##define PROBE_OFFSET_WIZARD_START_Z 4.0~~	~~#define PROBE_OFFSET_WIZARD_START_Z 4.0~~	
			~~#define BABYSTEPPING~~	~~#define BABYSTEPPING~~	
Temperature Sensor Type Selection	Most 3D printer components supplied online comes with EPCOS 3950 NTC Thermistor sensors (especially the heatbed). Hotend sensors may vary. This allows you to select your NTC thermistor model for better accuracy.	Marlin/ Configuration_adv.h	~~#define TEMP_SENSOR_0 1~~	~~#define TEMP_SENSOR_0 1~~	Heatbeds often use 3950 EPCOS NTC Thermistors, so leave it be in the config. For your hotend, however, I suggest checking your supplier's page or datasheet to see what thermistor was used. To know which value to input do a CTRL+F on “@section temperature” a table will show the different Thermistor models with their corresponding part code for the config.
			~~#define TEMP_SENSOR_BED 1~~	~~#define TEMP_SENSOR_BED 1~~	

					~~My Casi MK3 uses the standard 3950s for both hotend and heatbed, so I left them all as “1”. My Casi MK2 uses a Volcano-CHC with a 104NT-4 Thermistors, on that machine my value is “5” for the hotend and “1” for the heatbed.~~
--	--	--	--	--	---

PID Calib
BL-touch Offset
MPC
XYZ Skew
Disable Stealth chop

```
#error "Enable USE_YMAX_PLUG when homing Y to MAX."  
#define USE_YMIN_PLUG  >  //#define USE_YMIN_PLUG  
//#define USE_YMAX_PLUG  >  #define USE_YMAX_PLUG
```

- If you are using the exact same parts or identical one to my 3DP build, feel free to just copy the config values as a baseline.
- Measurements are often in units of mm
- Take note the formulas I used are simplified. To learn more about it, I recco reading prusa’s guide and calculator page. I wrote this guide specifically for my video tutorial (link), kindly watch the video first to get the full context of my notes.
- If ctrl+f in VScode cant find that specific line, try just copying the just variable name, sometimes spaces affects the search. I am using Marlin 2.1.2.2 when I was writing this guide. Marlin developers sometimes make slight changes to the variable names in newer firmware release updates. In cases where a newer version of Marlin is being configured and the variable from this guide can’t be found, kindly google that variable name or seek help in forums.

PARAMET ER	DESCRIPTION	WHERE TO FIND	LINE OF CODE TO MODIFY	MODIFIED LINE OF CODE	INSTRUCTION & NOTES
PID Calibration		Marlin/ Configuration.	#define DEFAULT_Kp 21.73 #define DEFAULT_Ki 1.54	#define DEFAULT_Kp 20.21 #define DEFAULT_Ki 1.91	PID calibration can be done afterwards and does not require

		h	#define DEFAULT_Kd 76.55	#define DEFAULT_Kd 53.36	changing the firmware configs directly. Run Ponterface or Octoprint to access the serial comms terminal. View Current PID Values: Type “M503” this will display all your 3DP’s configs including the PID values (Echo: M301 for hotend and M304 for heatbed). Feel free to copy it in a notepad for comparisons later. Run All Cooling Fans: Type “M106 S255” to enable all fans to take the cooling effect into account. PID Tune Hotend: Type “M303 E0 S230 C5”.... S230 is the temp at which we are tuning the printer for and C5 means 5 cycles of testing. When finished, it will return the PID values required for tuning. Copy it in notepad! Input PID Values For
			#define DEFAULT_bedKp 41.78 #define DEFAULT_bedKi 7.32 #define DEFAULT_bedKd 158.93	#define DEFAULT_bedKp 135.75 #define DEFAULT_bedKi 26.11 #define DEFAULT_bedKd 470.59	

					Hotend: Copy those acquired values for the hotend and type “M301 P<val> I<ival> D<val>”. Val refers to the individual PID values respectively. PID Tune Heatbed: Type “M303 E-1 S95 C5”. It’s just similar to the hotend PID tuning, but this time it’s for the heatbed. When finished, it will return the PID values required for tuning. Copy it in notepad! Input PID Values For Hotend: Copy those acquired values for the hotend and type “M304 P<val> I<ival> D<val>”. Val refers to the individual PID values respectively. Save The Settings: Type “M500”
Skew Calibration					